PROBLEM STATEMENT :

Assignment -3: HDR Imaging

- Objective – Implement HDR imaging algorithm
- Task – Implement HDR imaging algorithm to merge and tone map 3 differently exposed LDR images for high contrast scene
  - Capture 3 differently exposed image of LDR scene from your phone. Preferably day light outdoor or indoor having both bright, shadow and low light scene (e.g. sample images shown below for just reference, do not use as input to your algorithm)
  - Exposure can be controlled from your phone in manual mode (I assume you guys know now ☺)
  - Implement both merge of 3 images and tone mapping to 8bit for display.
- Report containing implementation details with observations

Github Link : https://github.com/yashbudhia/emmetra

We need to take 3 LDR images from our phone at different exposure times Input images :

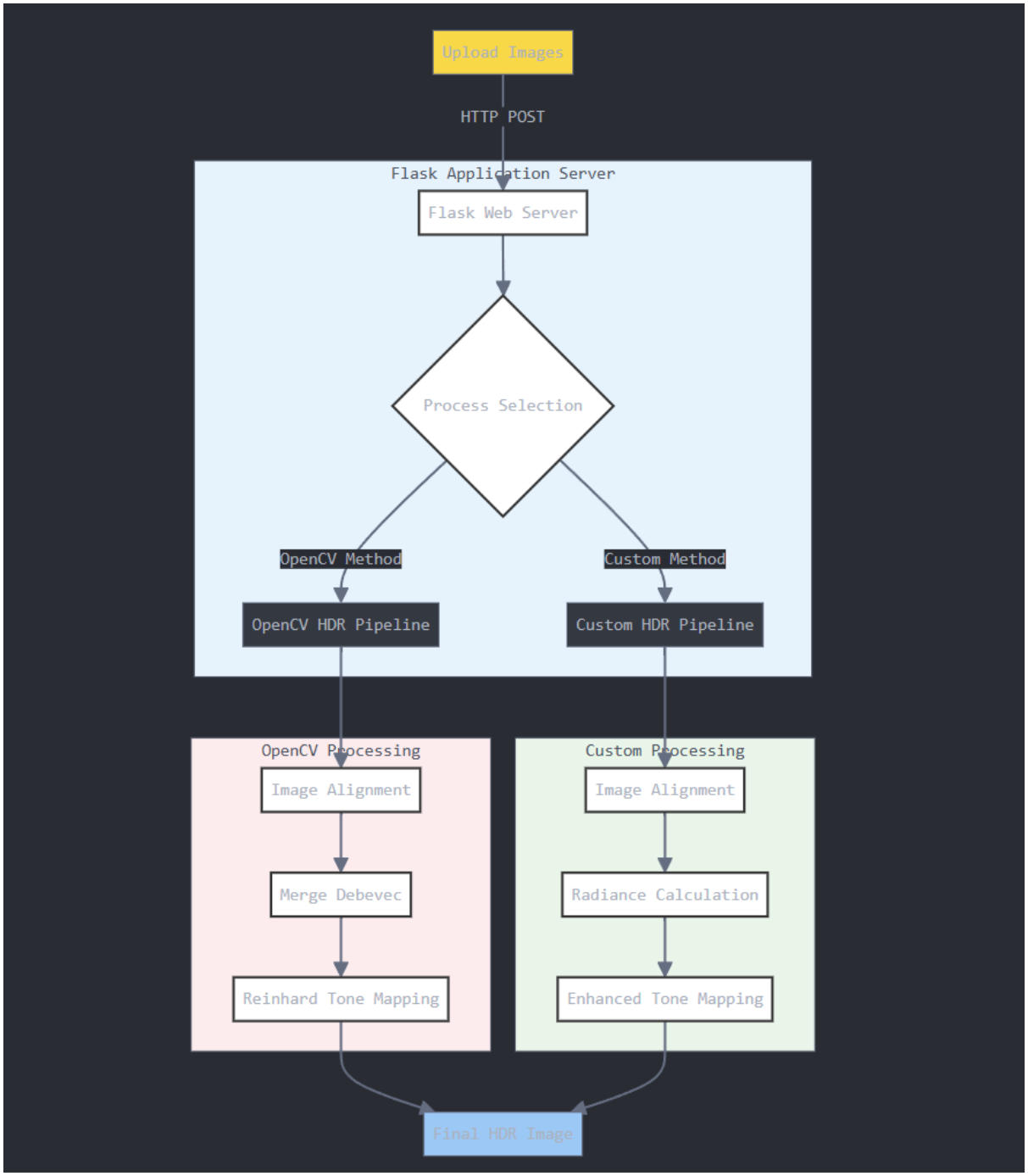
Our Approach –

- Input Layer:
  - User uploads three images (low, medium, high exposure)
- Server Layer:
  - Flask web server receives the images
  - Process selection between OpenCV and Custom methods
- Processing Pipelines:
  - OpenCV Pipeline (red background):
    - Image alignment
    - Merge Debevec algorithm

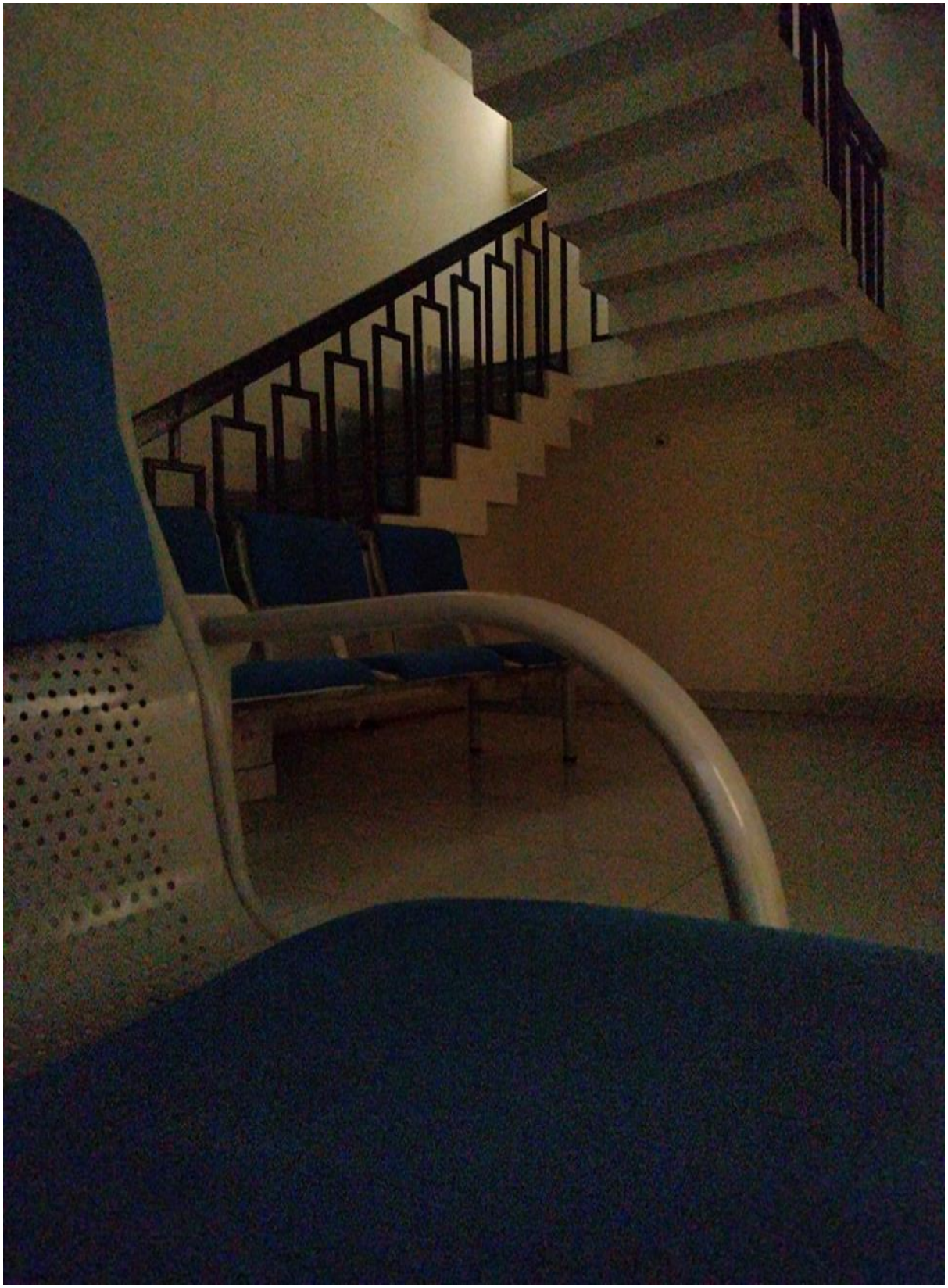
    - Reinhard tone mapping
- Custom Pipeline (green background):
    - Image alignment
    - Advanced radiance calculation
    - Enhanced tone mapping

- Output Layer:

    - Final HDR image generation



Input image with Low exposure time



Input image with Medium exposure time

Input image with High exposure time

We developed a user interface for users to upload the images.

Image Alignment: Aligning all the images is important for proper implementation of hdr algorithm

- ★ **Feature Detection**: Key features in the images are detected using the AKAZE feature detector.
- ★ **Feature Matching**: Features from the lower-exposure image (reference) are matched to features in the other images.
- ★ **Homography Estimation**: A homography matrix is estimated using the RANSAC algorithm to align each image to the reference image.
- ★ **Image Warping**: The other images are warped to match the perspective of the reference image based on the homography matrix.

Extracting Exposure Time:

We obtain exposure time from images by leveraging each image's metadata. This metadata, stored as EXIF data, typically contains camera settings, including exposure time. By accessing and reading this metadata, we can extract the precise shutter speed used for each image.

- ★ **Accessing Metadata:** Each uploaded image is scanned for embedded EXIF metadata, which often includes exposure settings from the camera.
- ★ **Extracting Exposure Time**: The function specifically searches for the "ExposureTime" field within the metadata, as this contains the shutter speed. If present, this field directly provides the exposure time as a fraction.

Initial approach :

Using OpenCV’s algorithm

OpenCV HDR Algorithm

The OpenCV method leverages OpenCV’s built-in HDR processing functions and is particularly suitable for creating HDR images by combining multiple exposures. The approach involves the following steps:

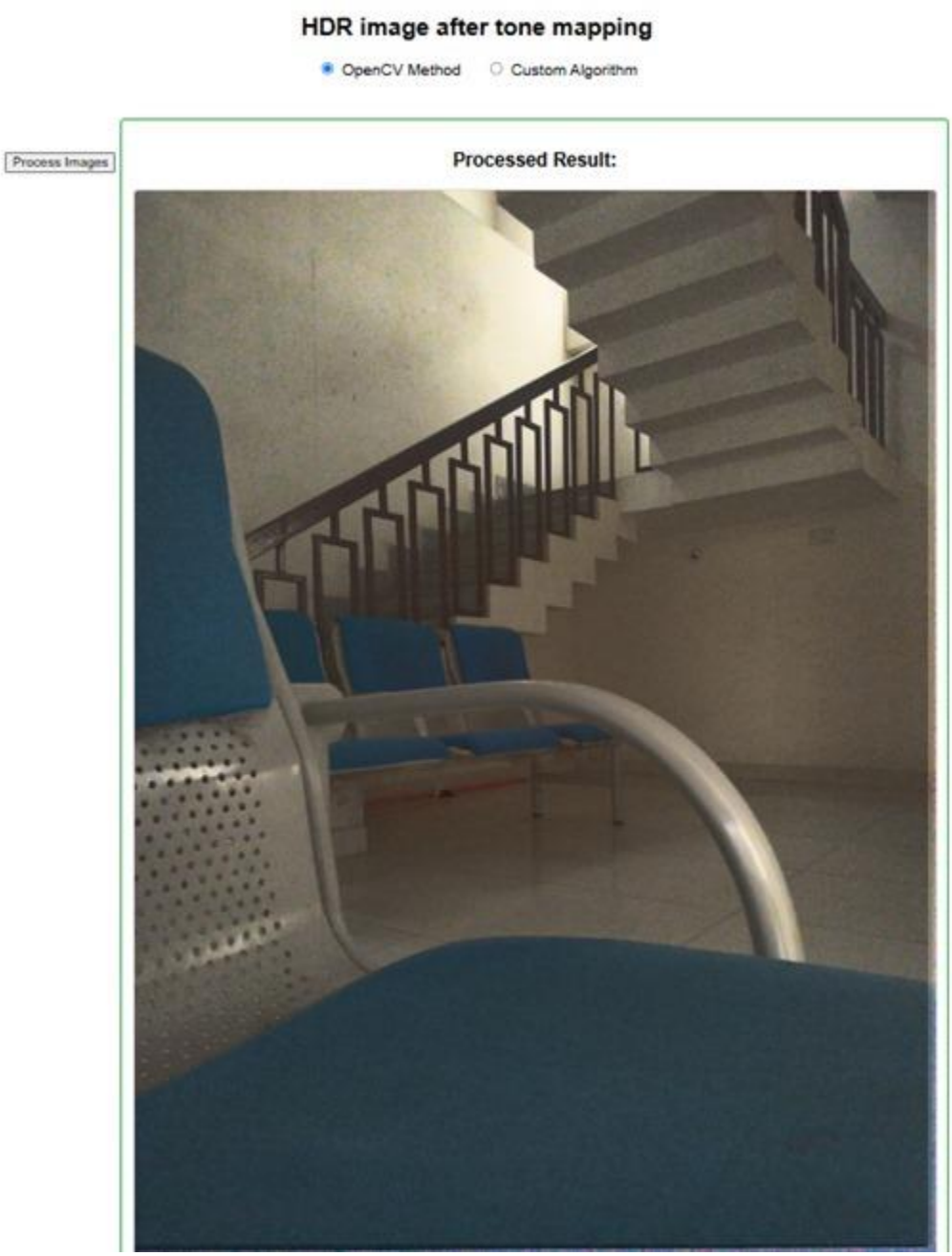
1.HDR Creation:

- ★ The aligned images are fed into OpenCV’s cv2.createMergeDebevec() function. This merges multiple exposures into a single HDR image by considering the camera’s exposure time for each shot, enhancing details in both dark and bright areas.

2.Tone Mapping:

- ★ Since HDR images have a high dynamic range, they must be compressed for display. OpenCV’s TonemapReinhard is used to perform tone mapping with a gamma adjustment (set to gamma=1.2 in this code).
- ★ Tone mapping converts the HDR image into an 8-bit Low Dynamic Range (LDR) image, which can be viewed on standard displays.

This is the Output Image using OpenCV algorithm:



Custom Approach

Custom HDR Algorithm

The custom HDR algorithm in this application is designed to handle HDR processing manually by calculating radiance values for each pixel based on the exposure times of the images.

Radiance Map Calculation

The Radiance Map Calculation step combines multiple images captured at different exposure levels into a single HDR radiance map. This is achieved by:

- ★ **Weighting Pixel Values:** A custom weighting function emphasises well-exposed pixels while down-weighting very dark or very bright pixels, helping to reduce noise and improve detail**.**
- ★ **Exposure Compensation:** Each image's pixel values are normalised based on its exposure time, allowing the HDR algorithm to properly balance brightness.
- ★ **Combining Images:** Using weighted summation, pixel intensities from all exposures are combined into a radiance map that represents the scene's true dynamic range.

This radiance map captures the full range of luminance levels, allowing for further tone mapping adjustments.

Tone Mapping

Tone Mapping compresses the high dynamic range of the radiance map into a low dynamic range suitable for standard displays. The tone mapping process includes:

- ★ **Luminance and Detail Extraction:** The luminance of the radiance map is separated into base and detail layers using a bilateral filter. This preserves fine details while smoothing overall brightness.
- ★ **Dynamic Range Compression:** The base layer's dynamic range is compressed to fit within displayable limits, while detail from the original HDR is retained.
- ★ **Colour Preservation and Saturation Boost:** Original colour ratios are preserved, with a controlled boost in saturation to make colours appear more vivid without distortion.
- ★ **Final Gamma Correction:** A gamma adjustment is applied to ensure natural-looking luminance on standard screens.

This is the Output Image using Custom HDR algorithm: